IN THE MATTER OF

U.S. Patent Application No. **10/046,912**

By Samsung Electronics Co., Ltd.

I, Jeong-hee Lee, an employee of Y.P.LEE, MOCK & PARTNERS of The Cheonghwa Bldg., 1571-18 Seocho-dong, Seocho-gu, Seoul, Republic of Korea, hereby declare that I am familiar with the Korean and English languages and that I am the translator of the priority document (Korean Patent Application No. 2001-38804) and certify that the following is to the best of my knowledge and belief a true and correct translation.

Signed this 9th day of March 2005.

Jeonghee Lee

ABSTRACT

[Abstract of the Disclosure]

A method of upgrading software and a network device for performing the same are provided so that the operation of a network device is not catastrophically affected by a failure occurring while software is being upgraded in a network environment. The method includes upgrading software through the network and checking whether at least one failure occurs during the upgrade, operating the network device based on an old version of the software used before the upgrade is performed when it is determined that at least one failure has occurred, and operating the network device based on a new version of the software to which the old version is upgraded when it is determined that a failure has not occurred. Accordingly, even if the upgrade of the software is not performed normally, the network device can attempt the upgrade of the software without being serviced or without using an external memory device.

[Representative Drawing]

FIG. 1.

SPECIFICATION

[Title of the Invention]

METHOD OF UPGRADING SOFTWARE IN NETWORK ENVIRONMENT AND NETWORK DEVICE FOR PERFORMING THE SAME

[Brief Description of the Drawings]

FIG. 1 is a block diagram of a network device according to the present invention.

FIG. 2 is a flowchart of a method of upgrading software according to the present invention.

[Detailed Description of the Invention]

[Object of the Invention]

[Technical Field of the Invention and Related Art prior to the Invention]

The present invention relates to a method of upgrading software in a network environment and a network device for performing the upgrade, and more particularly, to a method of upgrading software, through which a network device can be protected when an unexpected failure occurs in a network or a network device during an upgrade of the software, and a network device for performing the same.

As network environments develop, the need of network service providers, network operators, and network device users to upgrade software for network devices increases. Network devices include mobile and portable telephones and personal computers that can transmit and receive data through a network.

However, when an unexpected failure occurs in a network device or a network during a software upgrade of the network device, the upgrade is not completed normally. Moreover, a catastrophic influence may be exerted on the network device.

In other words, when a failure occurs in a network while a new version of software is being downloaded through the network to upgrade the software of a network device, the new version of the software cannot be downloaded normally. Here, the problem can be overcome by trying to download the new version again.

However, when an unexpected failure occurs in a network device while the new version of software is being stored in a storage area, from which the old version of the software has been erased in the network device, to upgrade the software, the network device cannot normally store the new version of the software. As a result, the network device cannot perform functions that use the software.

In the case where software to be upgraded does not greatly influence driving of a network device, even when an unexpected failure occurs in the network device as described above, a problem can be prevented by connecting an interface block provided in the network device to an external memory device and newly downloading the software. However, in this case, it is annoying that the external memory device must be managed to always store the new version of the software or to store at least the version of the software which is currently stored in the network device.

In the case where software to be upgraded greatly influences driving of a network device, as in an operating system (OS), when an unexpected failure occurs in the network device as described above, the network device is catastrophically affected and does not operate at all. Here, the problem cannot be solved by using an external memory device only, as described above. A corresponding chip or the network device must be replaced.

[Technical Goal of the Invention]

The present invention provides a method of upgrading software, through which the operation of a network device is not catastrophically affected by a failure occurring while software is being upgraded in a network environment, and a network device for performing the same.

The present invention also provides a method of upgrading software, through which a network device can operate normally even if software upgrade is not accomplished normally in a networked environment due to a failure, and a network device for the same.

According to an aspect of the present invention, there is provided a method for upgrading software of a network device through a network, the method including the steps of upgrading software through the network and checking whether at least one failure occurs during the upgrade, and operating the network device based on an old version of the software used before the upgrade is performed when at least one failure has occurred and operating the network device based on a new version of the software to which the old version is upgraded when a failure has not occurred.

The upgrading of software may include the steps of downloading the new version of the software through the network, copying the old version of the software stored in an area of the network device to another area of the network device, erasing the old version of the software from the area of the network device, and storing the new version of the software in the area where the old version of the software was stored.

The failure may be a failure in the network device which is checked during the erasing and storing steps.

According to another aspect of the present invention, there is provided a network device capable of upgrading software through a network, including monitoring means for monitoring at least one failure while software is being upgraded; a first memory for storing software necessary for operating the network device; a second memory for storing information transferred through the network; a controller for performing control to store information, which is downloaded through the network to upgrade the software, in the second memory, and store an old version of the software in an empty area of the first memory before the old version of the software stored in the first memory is upgraded with the information stored in the second memory; and a decoder for selecting a memory, which is used for upgrading the software, between the first memory and the second memory according to a control signal received from the controller and the result of monitoring received from the monitoring means, and setting an address.

According to still another aspect of the present invention, there is provided a network device capable of upgrading software through a network, including monitoring means for monitoring whether at least one failure occurs while software is being upgraded; a first memory for storing software necessary for operating the network device; a second memory for storing data necessary for operating the network device; a third memory for storing information transferred through the network; a controller for performing control to store information, which is downloaded through the network to upgrade the software, in the third memory, and store an old version of the software in an empty area of the second memory before the old version of the software stored in the first memory is upgraded to the information stored in the third memory; and a decoder for selecting a memory, which is used for upgrading the software, according to a control signal received from the controller and the result of monitoring received from the monitoring means, and setting an address.

[Structure and Operation of the Invention]

FIG. 1 is a block diagram of a network device according to the present invention. Referring to FIG. 1, a network device according to the present invention includes a monitoring unit 100, a network interface unit 104, a chip selection/address decoder 105, a controller 106, a conditional access system (CAS) 107, a flash memory 108 for storing code data, a non-volatile memory 109 for storing data, and a system memory 110 for storing execution data.

The monitoring unit 100 monitors at least one failure while software of a network device is being upgraded. The monitoring unit 100 includes a watchdog monitor 101, a power failure monitor 102, and a network connection monitor 103.

The watchdog monitor 101 monitors generation of a clock signal through which the operation of the network device can be monitored to monitor whether the network device operates normally, and sends the monitoring results to the chip selection/address decoder 105.

The power failure monitor 102 monitors whether a power voltage supplied from a power supply (not shown) within the network device drops below a predetermined level, and sends the monitoring result to the chip selection/address decoder 105.

The network connection monitor 103 monitors the state of data transmitted and received through the network interface unit 104, and sends the monitoring results to the chip selection/address decoder 105.

The network interface unit 104 performs interfacing to allow data communications between a party, which provides the new version of software to be upgraded through a network (not shown), and the network device shown in FIG. 1. In particular, the network interface unit 104 can be realized by decoding received software and checking received data for errors. Data error checking can be performed by way of checksum.

The CAS 107 transmits data and receives data between the party providing software through the network and the network device shown in FIG. 1 and performs authentication of the network device for a software upgrade. In other word, the CAS 107 verifies whether the network device has authority to upgrade corresponding software. Here, upgrading of software can be performed according to the determination of a software provider or at the request of a network device user.

The flash memory 108 for code data stores software such as an operating system (OS) necessary for operating the network device. When software stored in the flash memory 108 is upgraded, if the flash memory 108 has an empty area, the old version of the software can be stored in the empty area through the operation of the chip selection/address decoder 105.

The non-volatile memory 109 for data stores data necessary for operating the network device. In particular, when software stored in the flash memory 108 is upgraded, the old version of the software that has been stored in the flash memory 108 can be stored in an empty area existing in the non-volatile memory 109 through the operation of the chip selection/address decoder 105.

The system memory 110 stores information transmitted through the network when the network device is driven.

The controller 106 controls the network device to be able to upgrade software. In other words, once the new version of software to be upgraded is downloaded through the network interface unit 104 and the CAS 107, the controller 106 sends a control signal to the chip selection/address decoder 105 so that the new version of the software can be stored in the system memory 110.

Accordingly, the chip selection/address decoder 105 sets a chip selection signal for the system memory 110 and an address indicating an area of the system memory 110 in which the software is to be stored so that the received new version of the software can be stored in a desired area of the system memory 110.

Next, the controller 106 sends a control signal to the chip selection/address decoder 105 so that the old version of the software stored in the flash memory 108 can be stored in an empty area of the flash memory 108 or in an empty area of the non-volatile memory 109.

Accordingly, when the old version of the software is stored in the empty area of the flash memory 108, the chip selection/address decoder 105 sets a chip selection signal and an address so that a path for storing can be set. In other words, the chip selection/address decoder 105 does not change the chip selection signal but changes the address. When the old version of the software is stored in the empty area of the non-volatile memory 109, the chip selection/address decoder 105 sets a chip selection signal and an address so that a path for storing can be set. In other words, the chip selection/address decoder 105 changes both the chip selection signal and the address. Here, when the start address of the empty area of the non-volatile memory 109 is the same as the start address of an area where the old version of the software is stored in the flash memory 108, the chip selection/address decoder 105 changes only the chip selection signal. Therefore, the old version of the software stored in the flash memory 108 is copied to another area.

The controller 106 also sends a control signal to the chip selection/address decoder 105 so that the new version of the software stored in the system memory 110 can be stored in an area where the old version of the software is stored in the flash memory 108, thereby upgrading the software. The area where the old version of the software is stored is an area where the original old version of the software is stored in the flash memory 108 before the copying is performed.

The chip selection/address decoder 105 sets a chip selection signal and an address in response to a control signal sent by the controller 106 so that the new version of software stored in the system memory 110 can be stored in the flash memory 108.

While software is being upgraded, the chip selection/address decoder 105 continuously monitors the result of monitoring provided from the monitoring unit 100. When a signal indicating a failure is received from the network connection monitor 103 while the new version of the software is being downloaded to the system memory 110, the chip selection/address decoder 105 is initialized.

When a signal indicating a failure is received from the watchdog monitor 101 or the power failure monitor 102 while information stored in the system memory 110 is being stored in the flash memory, the chip selection/address decoder 105 sets a chip selection signal and an address to select a memory necessary for restarting the network device based on the old version of the software.

FIG. 2 is a flowchart of a method of upgrading software according to the present invention.

In an initial state in step 201, when the network connection between a software provider and a network device as shown in FIG. 1 is set in step 202, the version of software to be provided by the software provider is compared with the version of the software provided in the network device in step 203. If the version of the software provided in the network device is not old, since upgrading of the software is not necessary, the operation goes back to step 201 so that the network device is maintained at the initial state.

If the version of the software provided in the network device is old, upgrading of the software is started in step 204. User authentication is performed in step 205. In other words, authentication is performed through data transmission between the software provider and the network device for user authentication, as described with respect to the CAS 107 of FIG. 1. The authentication is accomplished by a typical well-known method.

If the user authentication ends in failure in step 205, the network device does not have authority to upgrade the software. The operation goes back to step 201. In contrast, if the user authentication ends successfully in step 205, the software is downloaded to the system memory 110 and a check is made as to whether a network disconnection has occurred and whether an error has occurred in the data of the received software in step 206. Data error checking for the received software can be performed by way of checksum.

When it is determined that at least one of a network disconnection and a data error in the received software occurred in step 206, the operation goes back to step 201, and upgrading of software is not performed.

In contrast, when it is determined that neither a network disconnection nor a data error occurred, a check is made as to whether downloading is completed in step 208. If it is determined that downloading is not complete, the operation goes back to step 206.

If it is determined that the downloading is completed in step 208, the old version of the software is copied to another area in step 209. Another area may be an empty area of the flash memory 108 or an empty area of the non-volatile memory 109, as described in FIG. 1.

In step 210, the software is upgraded to a new version, and it is checked whether a failure such as a power failure or system hang-up occurs in the network device during the upgrading.

Here, for a software upgrade, a path to an area in which the old version of the software is stored is changed to a path to the area to which the old version has been copied. In other words, when the old version has been copied to an empty area of the flash memory 108, the chip selection/address decoder 105 sets a chip selection signal and an address to designate the empty area of the flash memory 108. Next, the original old version of the software is erased from the flash memory 106, and then the new version of the software stored in the system memory 110 is recorded to the area from which the original old version is erased in the flash memory 106.

While the monitoring unit 100 is checking whether a failure such as a power failure or system hang-up occurs in the network device during the upgrading, when it is determined that at least one failure has occurred in the network device in step 211, the network device is restarted based on the old version of the software in step 212. Then, the operation returns to step 201.

In contrast, when it is determined that no failures have occurred in the network device in step 211, a check is made as to whether the software upgrade is completed in step 213. When it is determined that the software upgrade is not completed, the operation returns to step 210.

In contrast, when it is determined that the software upgrade is completed in step 213, the network device is restarted based on the new version of the software in step 214. Then, the operation returns to step 201.

[Effect of the Invention]

According to the present invention, while software is being upgraded to a new version through a network, if a sudden failure such as hang-up or latch-up of a network device or a failure in power occurs before the software upgrade is completed normally, the network device is restarted based on the software of an old version so that the network device can attempt the upgrade of the software again without being serviced or without using an external memory device.

What is claimed is:

1. A network device capable of upgrading software through a network, comprising:

monitoring means for monitoring at least one failure while software is being upgraded;

a first memory for storing software necessary for operating the network device;

a second memory for storing information transferred through the network;

a controller for performing control to store information, which is downloaded through the network to upgrade the software, in the second memory, and store an old version of the software in an empty area of the first memory before the old version of the software stored in the first memory is upgraded with the information stored in the second memory; and

a decoder for selecting a memory, which is used for upgrading the software, between the first memory and the second memory according to a control signal received from the controller and the result of monitoring received from the monitoring means, and setting an address.

2. The network device of claim 1, wherein the controller provides a control signal to the decoder to store the old version of the predetermined software to the empty area of the first memory, erase the old version of the software stored in an original area of the first memory, and store the information stored in the second memory to the original area of the first memory.

3. The network device of claim 1, wherein the monitoring means monitors whether a failure including one of a power failure and hang-up occurs in a network device.

4. The network device of claim 1 or 3, wherein the monitoring means monitors whether a failure in the network occurs.

5. The network device of claim 1, wherein when the decoder receives a signal indicating that at least one failure has occurred from the monitoring means while the software is being upgraded, the decoder returns to the initial state of the network device.

6. The network device of claim 5, wherein when the failure occurs while the old version of the software is being upgraded after the old version of the software is stored in the empty area of the first memory, the decoder operates so that the network device can be restarted based on the old version of the software.

7. A network device capable of upgrading software through a network, comprising:

monitoring means for monitoring whether at least one failure occurs while software is being upgraded;

a first memory for storing software necessary for operating the network device;

a second memory for storing data necessary for operating the network device;

a third memory for storing information transferred through the network;

a controller for performing control to store information, which is downloaded through the network to upgrade the software, in the third memory, and store an old version of the software in an empty area of the second memory before the old version of the software stored in the first memory is upgraded to the information stored in the third memory; and

a decoder for selecting a memory, which is used for upgrading the software, according to a control signal received from the controller and the result of monitoring received from the monitoring means, and setting an address.

8. A method for upgrading software of a network device through a network, the method comprising the steps of:

upgrading software through the network and checking whether at least one failure occurs during the upgrade; and

operating the network device based on an old version of the software used before the upgrade is performed when at least one failure has occurred and operating the network device based on a new version of the software to which the old version is upgraded when a failure has not occurred.

9. The method of claim 8, wherein the upgrading of software comprises the steps of:

downloading the new version of the software through the network;

copying the old version of the software stored in an area of the network device to another area of the network device;

erasing the old version of the software from the area of the network device; and

storing the new version of the software in the area where the old version of the software was stored.

10. The method of claim 9, wherein the failure is a failure in the network device which is checked during the erasing and storing steps.

11. The method of claim 8, wherein the failure comprises a power failure in the network device or hang-up of the network device.

12. The method of claim 8 or 11, wherein the failure comprises a network failure.

FIG. 1

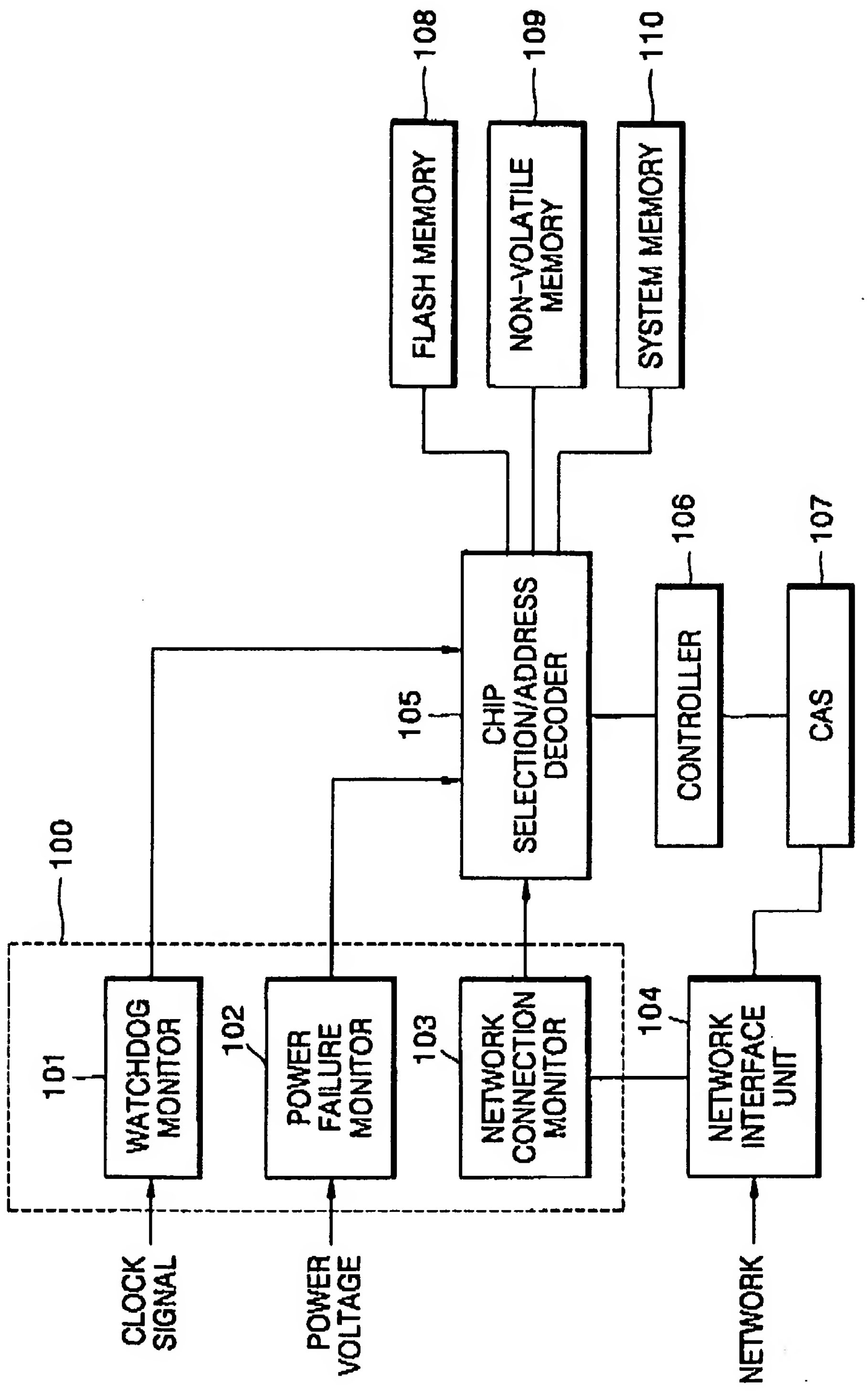
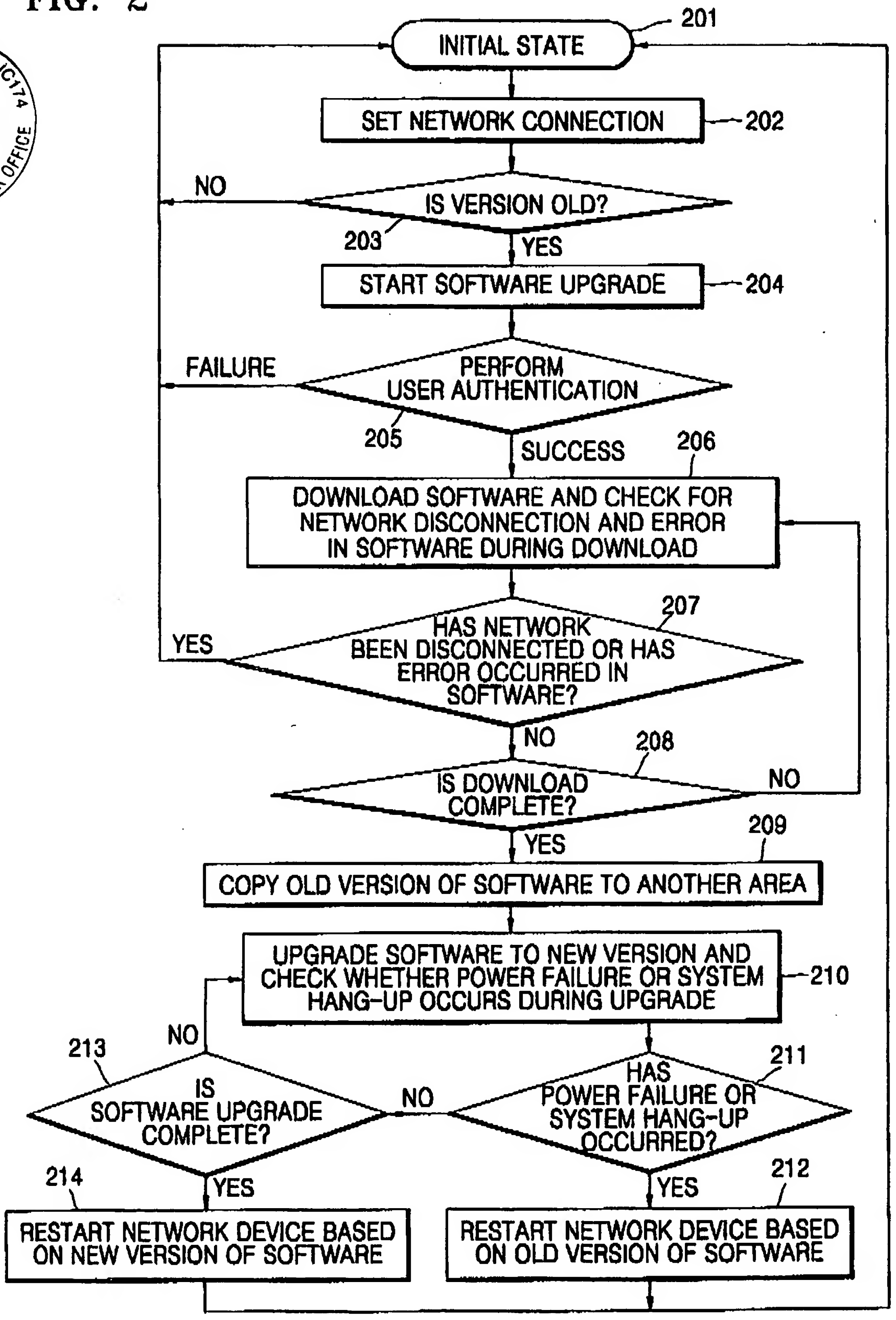

108
109
110
FLASH MEMORY
NON-VOLATILE MEMORY
SYSTEM MEMORY
106
107
105
CHIP SELECTION/ADDRESS DECODER
CONTROLLER
CAS
100
101
WATCHDOG MONITOR
102
POWER FAILURE MONITOR
103
NETWORK CONNECTION MONITOR
104
NETWORK INTERFACE UNIT
CLOCK SIGNAL
POWER VOLTAGE
NETWORK

FIG. 2


INITIAL STATE
201
SET NETWORK CONNECTION
202
IS VERSION OLD?
203
NO
YES
START SOFTWARE UPGRADE
204
PERFORM USER AUTHENTICATION
205
FAILURE
SUCCESS
206
DOWNLOAD SOFTWARE AND CHECK FOR NETWORK DISCONNECTION AND ERROR IN SOFTWARE DURING DOWNLOAD
207
HAS NETWORK BEEN DISCONNECTED OR HAS ERROR OCCURRED IN SOFTWARE?
YES
NO
208
IS DOWNLOAD COMPLETE?
NO
YES
209
COPY OLD VERSION OF SOFTWARE TO ANOTHER AREA
UPGRADE SOFTWARE TO NEW VERSION AND CHECK WHETHER POWER FAILURE OR SYSTEM HANG-UP OCCURS DURING UPGRADE
210
213
NO
IS SOFTWARE UPGRADE COMPLETE?
NO
211
HAS POWER FAILURE OR SYSTEM HANG-UP OCCURRED?
214
YES
YES
212
RESTART NETWORK DEVICE BASED ON NEW VERSION OF SOFTWARE
RESTART NETWORK DEVICE BASED ON OLD VERSION OF SOFTWARE